**UML Project - Practo Health**

**Team members:** Hardik Jhaveri, Jay Sheth, Prerak Sheth

**INTRODUCTION:**

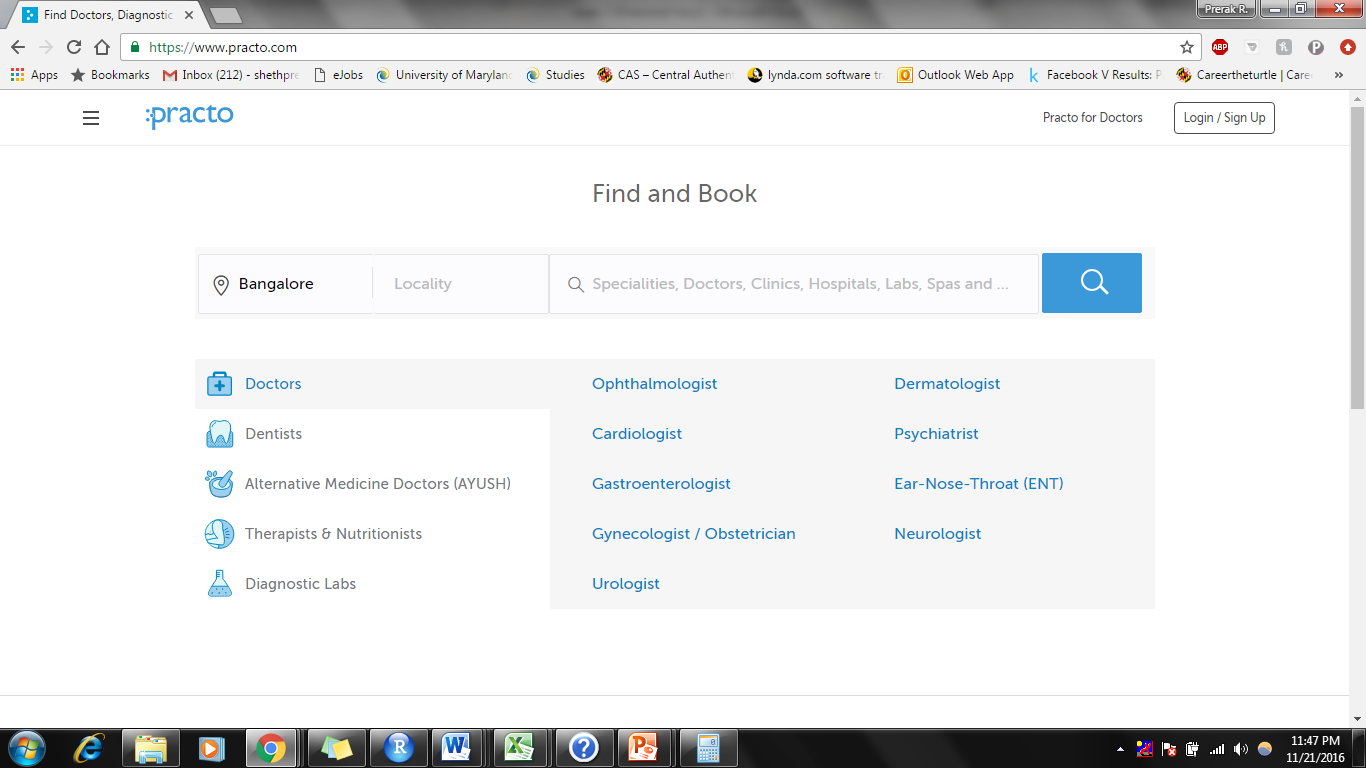
The system that we have chosen for our UML project is a health care website known as Practo. The link to the website is:<https://www.practo.com/>. Our reason to choose Practo was primarily our interest in the healthcare industry and how information systems are conveniently changing the way healthcare functions for both its users and owners.

Practo Health was founded in 2009 in Bangalore, India by two college friends Shashank ND and Abhinav Lal. The idea came to Abhinav’s mind when he was trying to find the right doctor to get a second opinion for his father's knee replacement surgery. He could not find the right medium to find someone who was skilled enough to be trusted with the second opinion. This led to the origin of Practo, a healthcare company which can be used by both, patients and doctors. Patients can use Practo to find, book appointments with doctors, and few doctors use Practo Ray a Saas software to manage their practice.

Practo can be accessed from the company’s website as well as the mobile applications on Apple’s iOS and Google’s Android OS. The website has multiple functions which mainly comprises of search for doctors, hospitals, labs, etc. One can manually enter the search location or enable GPS to detect current location. Additionally, there is a link to look up for a doctor to consult with by chat which is available 24 hours. The above functions can be used as a guest user without logging in, however there is a login function available at the main screen. Logging in provides the user with information about search history, recommended doctors and also lets a person store medical records like blood tests, prescriptions etc.

*The Practo System:*

The Practo system consists of three subsystems, i.e, the website, the mobile application and the Practo Ray. The Practo Ray is a software which is catered to doctors for easy management of their appointments, billing, analysis and reports. The mobile application for Practo is available on both Android and IOS. In this project we are focusing on the Practo website system and its functionalities. Below is the screenshot of how the system looks.



**DELIVERABLES:**

1. A detailed use case diagram of the system as defined by the system boundary.

2. A use case definition of one of the non-trivial use cases from the system’s use case diagram.

3. An activity diagram of one of the non-trivial use cases from the system’s use case diagram.

4. A sequence diagram of one of the scenarios from the chosen activity diagram.

5. A communication diagram of one of the scenarios from the chosen activity diagram.

6. A class diagram of the system as defined by the system boundary.

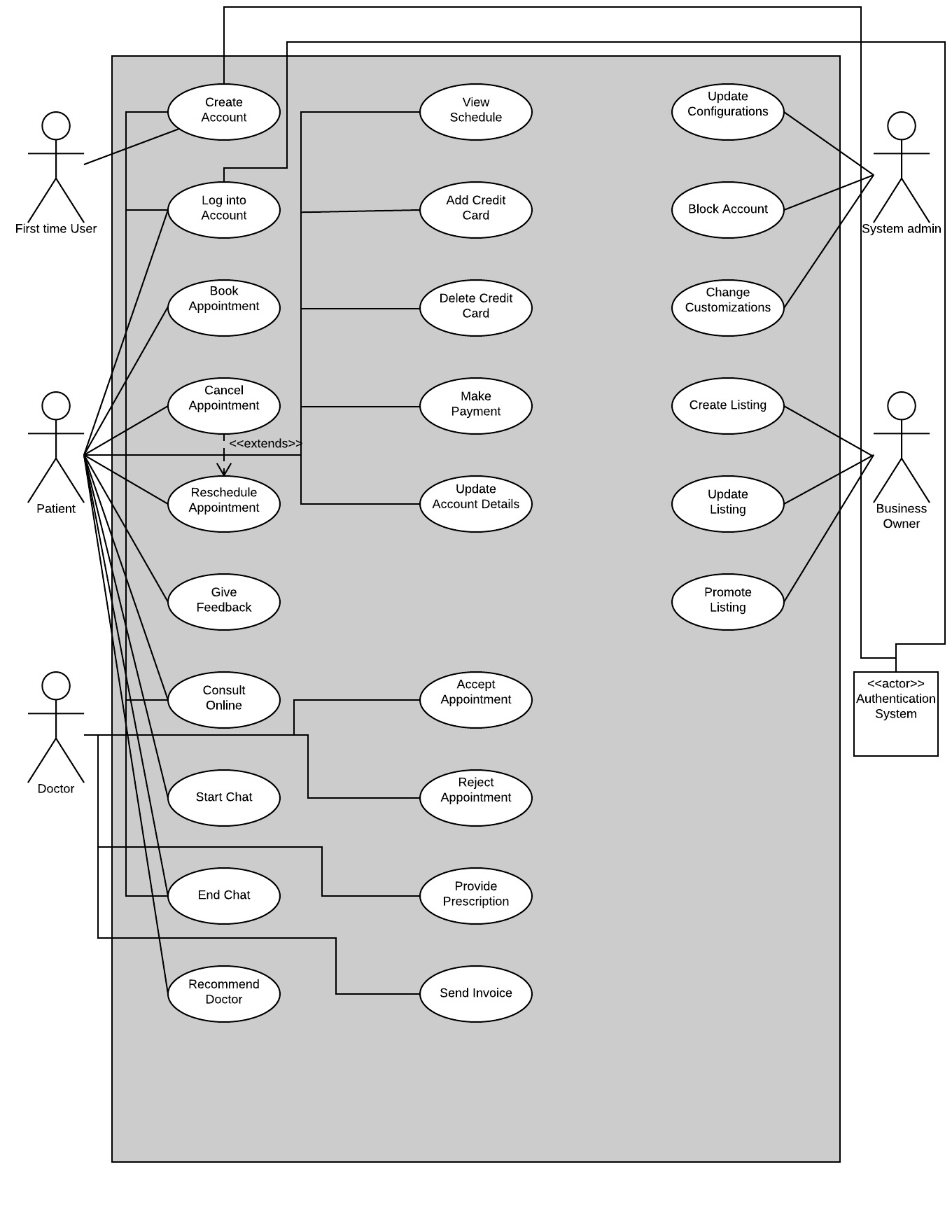
A report introducing the system, the challenges faced during the modeling process, how the challenges were addressed, learning as a result of working on the project.

**SCOPE:**

For the purpose of modelling, defining the scope of our project is of prime importance since every model should have a defined boundary. For this project, we will consider only the main Practo website. This website can primarily have four types of users namely the patient (customer), doctor system admin and business owners. The website can be used by the user to create and access their account to check blood bank/hospital listings posted by the business owners or find doctors who have an active account on the website. The users can also schedule/re-schedule an appointment and write feedbacks on their visit with doctors and blood banks. The doctors can accept or reject the appointment and also send an invoice to the user. The user can make payments for these invoices.

Our project will not cover any aspects of the Practo mobile applications or the Prato Ray for Doctors module. The diagrams below are only restricted to the scope highlighted above.

**USE CASE DIAGRAM**



**Figure 1: Use Case Diagram**

The Practo Healthcare system serves to four types of users, patients or visitors, doctors, business owners and system admin. The system fulfils multiple user requirements. Every Use Case captures a piece of functionality that the system provides. These Use Cases and their interaction with the actors make a Use Case Diagram which is essentially related to all of the other diagrams in the model. The diagram above just shows what the system is supposed to do and not how it is supposed to do. The diagram comprises of Actors, Use Cases, Communication Lines and System Boundaries.

*Actors and its Use Cases*

The actors comprises of the users of the system and the other relevant components of the system. Following are the actors represented in the diagram above and a short description for each.

1. First Time User: When any person visits the website for the first time and does not own an account, there is nothing the user can do apart from creating a new account. Practo doesn’t allow users to access its services without creating an account. This user then creates an account. The first time user has access to following Use Cases:
2. Create Account: This allows the first time user to create an account when they access the website for the first time.
3. Patient: The patient is a user who has an existing account or just created an account. This person has access to various functionalities of the system. The patient can also interact with the doctor through various functionalities. The patient can access the following Use Cases:
4. Log in to Account: This allows the patient to log into an existing account and access functionalities of the system.
5. Book Appointment: This allows the patient to book an appointment with the doctor.
6. Cancel Appointment: This allows the patient cancel the previously booked appointment. Cancel Appointment extends Reschedule Appointment.
7. Reschedule Appointment: This Use Case helps the patient reschedule an appointment. The user can also cancel appointment through this use case.
8. Give Feedback: The patient can provide a Feedback about the services provided by the doctor.
9. Consult Online: This allows the patient to consult a doctor online
10. Start Chat: This allows the patient to start a chat with the doctor.
11. End Chat: The patient can also end an ongoing chat with the doctor.
12. Recommend Doctor: This lets one patient to recommend a doctor to other patients.
13. View Schedule: This lets the patient view their own schedule of the booked appointments
14. Add Credit Card: A patient can also Add a card to their account to expedite payments
15. Delete Credit Card: Old or unused cards can be deleted from the account by the patient
16. Make Payment: The patient can pay the doctor online using this use case.
17. Update Account Details: The patient can update various aspects of their own account using this use case.

3. Doctor: A doctor can be a first time visitor or a signed up doctor. The doctor can perform various activities including a few that involves interaction with patients.

1. Create Account: The doctor can create a new account in spite of having one account since one doctor can have multiple areas of expertise and also have multiple clinics.
2. Log into Account: A doctor can log into an existing account and access different functionalities available for use.
3. End Chat: A doctor is not allowed to start a chat with the user. However, they can end an ongoing chat.
4. Consult Online: When the patient consults a doctor online, the doctor is at the receiving end and hence needs access to this functionality.
5. Accept Appointment: A doctor has the freedom to accept the appointment once the patient has already requested for one
6. Reject Appointment: Doctor can also reject a patients requested appointment
7. Provide Prescription: The doctor can provide an online prescription post the patients visit
8. Send Invoice: The doctor can send an invoice to the patient for the visit or consulting.

4. System admin: The system administrator is the super-user of the system and has access to the system from the backend. The following are the functionalities a system admin has access to:

1. Update Configuration: The configurations of the website can be updated by the System admin on a ‘need to’ basis.
2. Block Account: The system admin can block a user or doctor account if required
3. Change Customizations: The system admin can change to customizations of the website to play with the look and feel of the website

5. Business Owner: Apart from lists of doctors, there can be other businesses such as hospitals and pathology labs on the website. It is essential for a representative of a business organization to have access to the website. The following are the use cases the business owner has access to:

1. Create Listing: A first time business owner can create a listing on the website
2. Update Listing: This allows the business owner to update existing listings on the website
3. Promote Listing: A business owner can do a paid promotion of the listing on the website.

5. Authentication System: The authentication system stores various aspects of the account for security of the system. The following are its use cases:

1. Create Account: As soon as the user/doctor creates an account, the data is stored in the authentication system for future reference
2. Log into Account: If a patient or doctor tries to log into an account, the system checks with the authentication system for previously stored parameters.

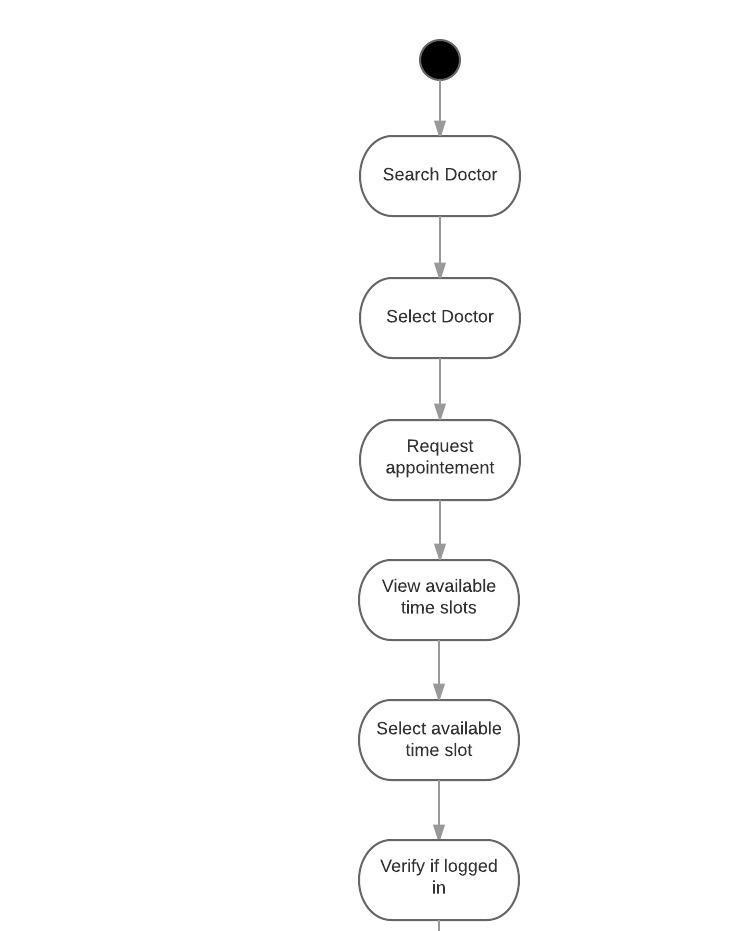
*Use Case Description:* A Use Case Description provides a text-based detail information about a use case. All of the use cases described above should ideally be defined using the use case description. However, below is a description of an essential use case ‘Create Account’ which gives an idea on how to describe the other use cases.

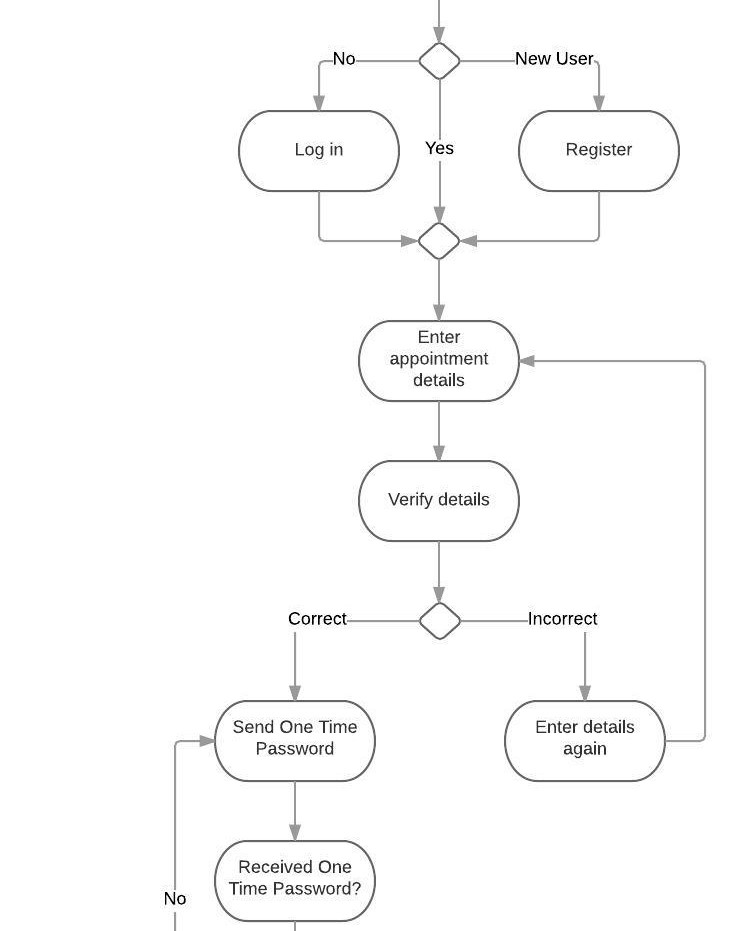
|  |  |  |
| --- | --- | --- |
| Use Case Name | Create Account | |
| Related Requirements | Requirement A.1 | |
| Goal in Context | First time users and doctors need to create a new account | |
| Preconditions | The users/doctor must have a valid email ID and mobile number for verification | |
| Successful End Condition | A new account for the user/doctor is created | |
| Failed End Condition | A new account for the user/doctor is not created | |
| Primary Actors | First time users and doctors | |
|  | Secondary Actors | Authentication System |
|  | Trigger | The first time user/doctor requests the system to create account |
| Main Flow | Step | Action |
|  | 1 | The first time user/doctor requests the system to create account |
|  | 2 | The first time user/doctor enters their details |
|  | 3 | The system verifies the email ID and phone number |
|  | 4 | A new account is created |
|  | 5 | Summary of the account is email to first time user/doctor |

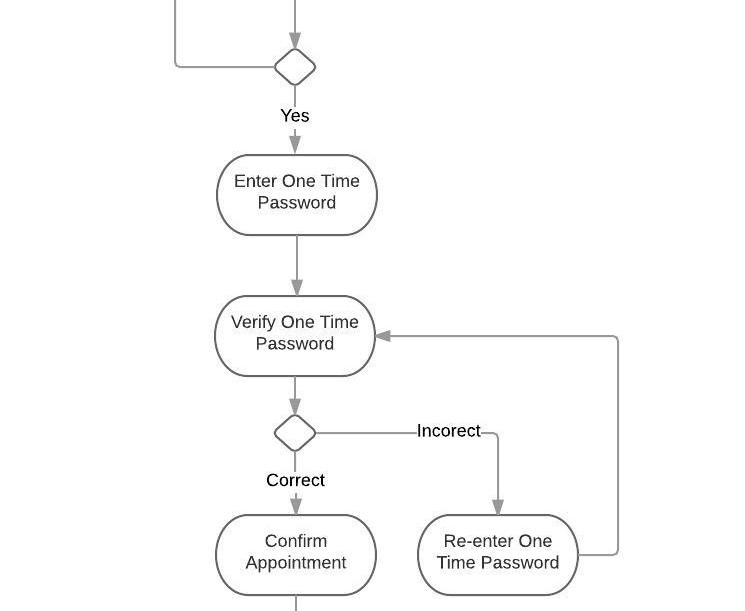
Following are the items listed in the use case description:

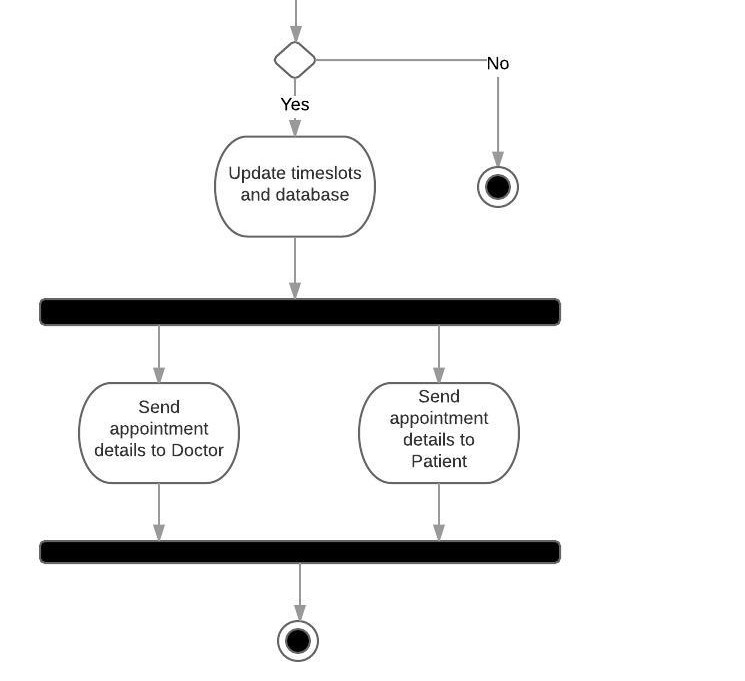
1. Use Case Name: This is just the english name of the use case
2. Related Requirements: This are the requirements section referenced to where all the implementation requirements are noted.
3. Goal in Context: This is the goal of using the use case
4. Preconditions: This are the conditions which need to be satisfied to be able to use the use case
5. Successful end condition: This is the ideal state that is desired to be achieved by using this use case
6. Failed End Condition: This is the condition that the system will be in if the successful end condition is not reached
7. Primary Actors: The actors that are the primary users of the system
8. Secondary Actors: The actors that are not the primary users but need access to the use case to enable execution of the system
9. Trigger: The event triggered that leads to execution of the use case
10. Main Flow: This describes the stages of the use cases normal execution.

**ACTIVITY DIAGRAM**









**Figure 2: Activity Diagram**

One of the salient features of Practo is that a patient can search specialized doctors and practitioners and book an appointment with them. This feature is available to patients at any time of the day. Through this important and effective solution provided by Practo, patients can confirm an appointment with a specialized doctor at the time which is most convenient to them as well as without any charge.

The activity diagram in our deliverable captures the entire process of a user who is looking to book an appointment with a doctor. This activity diagram covers all the possible scenarios possible to understand the book appointment use case of this system.

The following details gives a full pictures of all possible scenarios:

1. Search and select a doctor

Any user of the website can either use the search bar to enter a specialized field of medicine to view doctors from that stream or can enter a doctor’s name. The user can also search by location of the doctor. The user will then need to select a doctor to book an appointment.

1. Request an appointment, view availability and select a timeslot

Once the user has selected the desired doctor, they must click on book an appointment for that particular doctor. On performing that action the user will get the availability of that doctor. The user must now select an available time according to their preference.

1. Log in or Register

Now the system requires authentication from the user’s side and asks the user to log in so the system can understand who is requesting an appointment. If the user has already logged in then it will continue to the next step. If the user hasn’t yet logged in then it will ask the user for his/her credentials and then continue. The system also allows to register if the user is a new user of the system.

1. Appointment details

Now the system will ask the user to enter appointment details such as name, email id, and mobile number of patient. The system takes these values and verifies if they are appropriate. If the system identifies that the mobile number or email id does not adhere to the ideal form then it will ask the user to enter the details again. If the details are appropriate then it will move on to the next step.

1. One time password

Practo uses an authentication system where it sends a password to the user on their mobile phone as a text message. This technique is used for accountability and assurance from the user’s side Once the user enters the appointment details, the system sends this one time password to the user and requires the user to enter this so it can verify if it a genuine user. However sometimes, users do not receive a text which contain this password and may have to request the system to send it again on their mobile number. The user then must enter this password to move on to the next step.

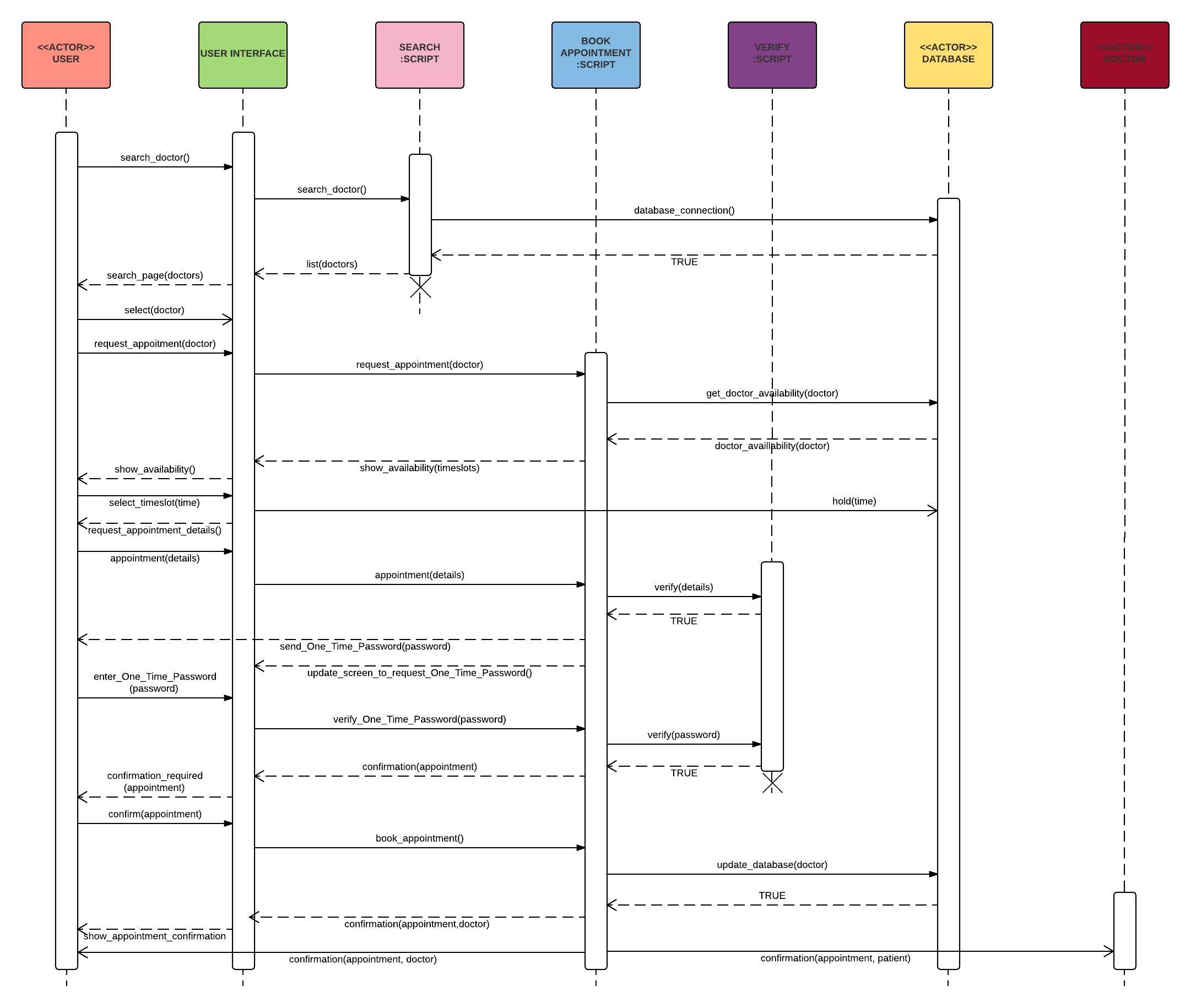
1. Confirm appointment

Once the system verifies the one time password which the user entered previously, the system knows that this is a genuine user. The system will now ask the user if they want to confirm the appointment. This is the final step the user has to take to book the appointment. If the user selects yes then the appointment is booked and if the user selects no then the user exits the entire process.

1. Update database and send confirmation to the doctor and patient

If the user selects yes and confirms the appointment then the system will update the database and the doctor’s schedule so that other users cannot book an appointment for the same time. Once the database is updated the system will send an email and text message notification to the doctor and the patient about the appointment with details of date and time.

**SEQUENCE DIAGRAM**

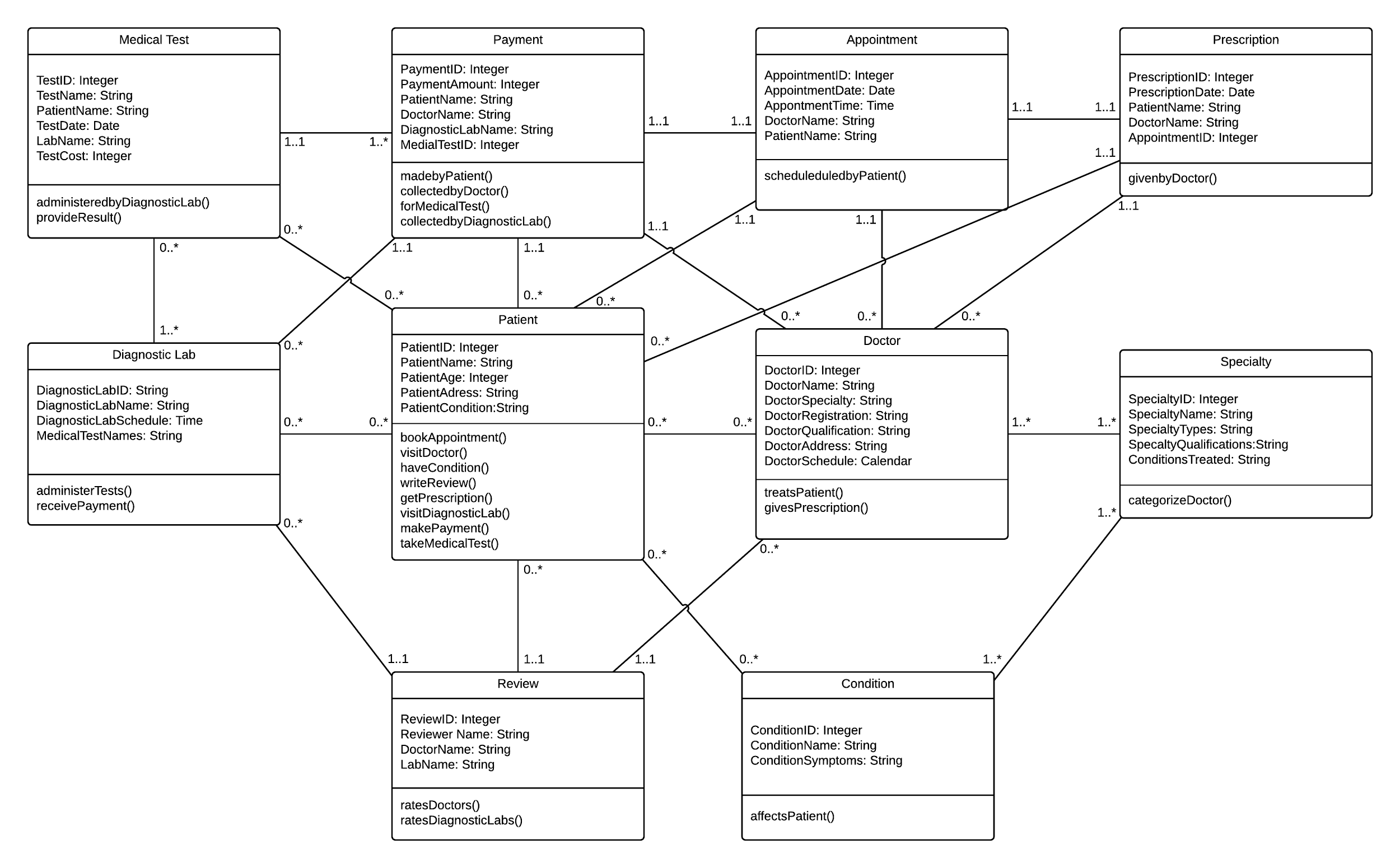


**Figure 3: Sequence Diagram**

We have chosen a scenario from the activity diagram where the user successfully books an appointment with the doctor The sequence of the events which will take place in this process are:

1. The user will search for a doctor based on specification, location, or name.
2. This will invoke the search script with the searching parameter.
3. The search script will connect to the database and retrieve values from the database.
4. The user will see the list of doctors on the user interface screen.
5. The user will now select a doctor on the user interface. This selection will not change the page but show more details of the doctor where there is a ‘request appointment’ button.
6. Now after pressing the request appointment, the book appointment script will connect to the database and retrieve available timeslots of the doctor.
7. The user will see available times of the doctor on the user interface. The user selects one of the available times. The system holds the time in the database till the process is completed.
8. Now the user fills in the details such as name, email and mobile number on the screen.
9. We are assuming that these are correct and the user is verified by the verify script.
10. Now we are assuming that the user will receive the one time password from the book appointment script directly on his/her cell phone whose number was given before. Then the screen will ask for the password.
11. Once the user receives the password, the user enters the password correctly on the screen.
12. The correct password will be verified by the verify script and will alert the book appointment script that it is a genuine user requesting an appointment.
13. Now the book appoint script sends a confirmation request to the user.
14. We are assuming that the user will accept the confirmation.
15. The book appointment screen will now update the database and the database which had held the time for the user will confirm the appointment and update.
16. Now the book appointment script will concurrently send a notification to the user and the doctor.

**CLASS DIAGRAM:**



**Figure 4: Class Diagram**

The class diagram consists of 10 classes as shown in the diagram. These are described below:

Patient: The patient class is the most important class as it is one of the central classes and has connections to a number of classes. It basically consists of attributes patients name and details regarding the patient. The patient class can carry out operations such as bookAppointment(), visitDoctor(), getReview().

Doctor: The Doctor class also has a number of connections. The attributes of the Doctor class provides details about the doctor such as the address and schedule. The doctor can carry out operations such as treatPatient() and givePrescription().

DiagnosticLab: This is the third component of the class diagram. The Practo portal also connects patients to diagnostic labs where they can visit to undergo any sort if medical test. Labs can perform operations such as administerTests().

MedicalTest: MedicalTest class depicts the particular test that would be administered on a patient by a lab. It consists of details of the tests.

Appointment: The Appointment class attributes provide details such as the time, patient’s name and the doctor’s name. It basically depicts a visit by a patient to a doctor.

Prescription: The prescription is provided by the Doctor to the Patient when he visits the Doctor.

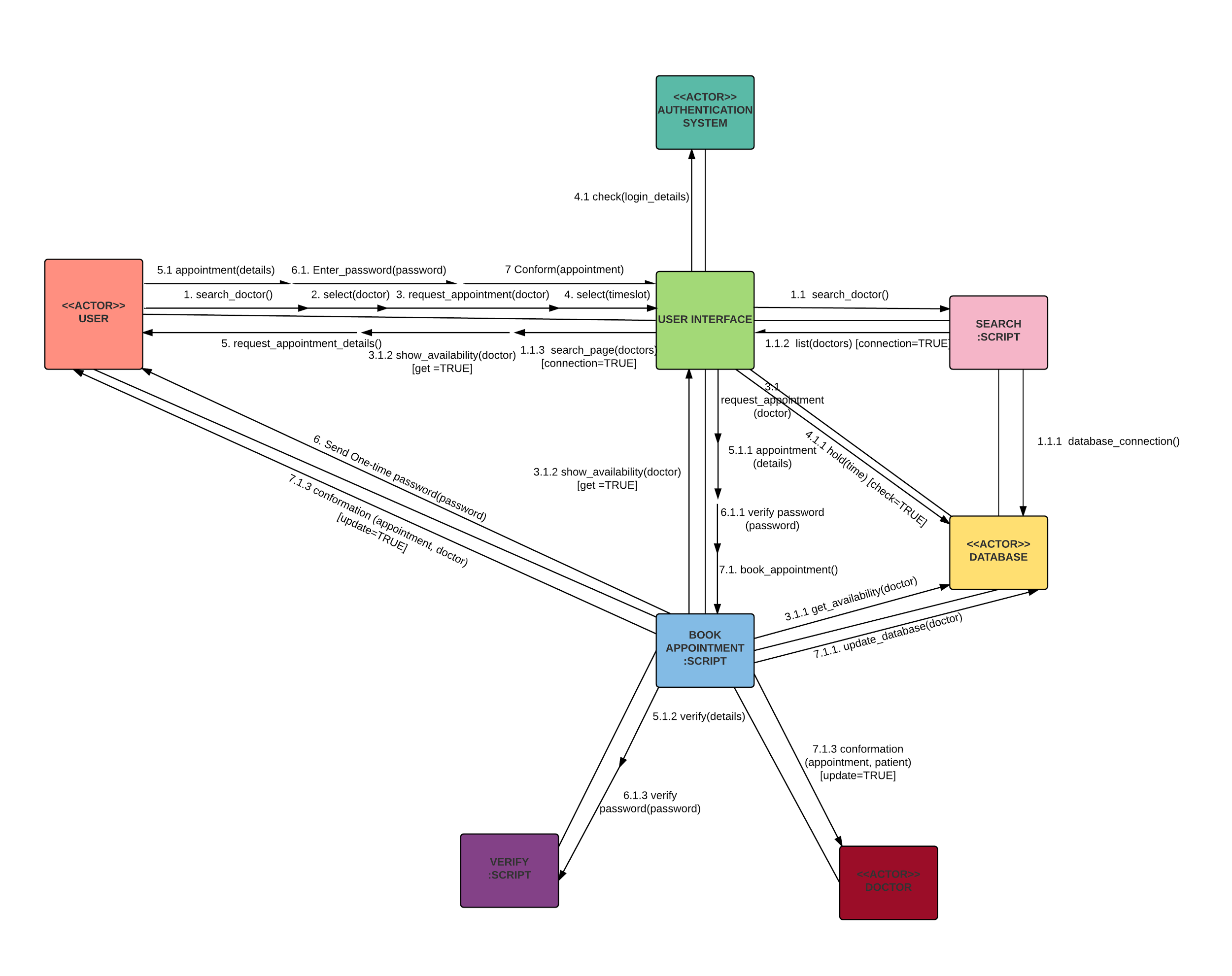
Payment: Payment class provides details about the Payment that the Patient makes to the Doctor or a Diagnostic lab for the services provided by them.

Condition: The Condition class basically describes the various conditions under which patients would visit a doctor. It has an attribute which would list a symptom related to that condition.

Specialty: The specialty class is basically a class which depicts the different specialty of doctors that can be accessed via the Practo portal. A specific doctor would be able to treat patients with certain conditions. Thus, it is connected to Doctor and Condition.

Review: The Patient can write review about the Doctors and the Labs that he/she would have visited.

**COMMUNICATION DIAGRAM:**



**Figure 5: Communication Diagram**

The communication diagram shows the communication between the different actors. Here we have considered the scenario where the user books an appointment. This diagram will add a new perspective to our system by showing new links between the actors like user, database, user interface, verify script, doctor, book appointment script and search script. Here we have shown the messages which are exchanged between these actors during the process of appointment booking.

**CHALLENGES AND SOLUTIONS:**

We faced a number of challenges while modeling the Practo system as it was a new and unique system. None of us had ever used this system before. Hence we first explored the entire system and then modeled it.

Practo is a huge website which connects users to not only doctors but also gymnasiums, spas, clinics, hospitals and nutritionist. Practo not only works for patients but also doctors and other business owners to track their business, schedules, payments, history and consulting. Hence the main challenge for us was to define the scope of this modeling project. One of the main challenge of this system was that is widely used in India, Brazil, Indonesia, Philippines and Singapore but has not be running in America. Therefore the registration process was difficult since it required a mobile number from those areas. While booking an appointment a one-time password was sent to the mobile which was difficult to acquire. Hence to understand the process of booking an appointment was challenging for us.

While making the diagrams many technical as well as non-technical details were to be addressed such as how the database session was created during making an appointment, payments and authentication system. Non-technical details like making an account and understanding the functions of the system was a major challenge.

We first thing we did was to define the scope of our project so we could perfectly understand the system and model it to very fine detail. We used our personal contacts from India so we could create an account on the website and actually book an appointment. This is how we learned about the entire process. The website had a help page which stated all the functionalities and behind the scene algorithms and mechanisms. The website also had an introductory video for new users to understand the system.

**LEARNINGS:**

This project enables the three of us learn numerous things in the space of technology specially a lot of industrial application when working in development and delivery. Coincidentally, all the three of us are interested to advance our careers as Business Analysts and I believe this project has contributed to expand our knowledge in this domain.

1. A standard for Software Development: UML is used as a standard language for modeling across the world in this domain and working on an extensive project like this has strengthened our foundation in using UML professionally. This enables smooth communication within and outside collaborative teams working on building a top-notch system. It lays a good boundary between stakeholders and system developers.
2. Reduction of costs: A standardized language like this can drastically help cut down on costs by ensuring smooth and quick communication. It also leads to lessen the development time required to build the product.
3. Avoids miscommunications: This language has a standard and easy format which helps us easily describe different aspects of the system which can give zero errors in communication. It has large visual elements to construct easy step by step diagrams and is usually easy to follow. These diagrams can also act as basis for negotiations and agreements on system requirements and costs. It is also a solid representation of what a system will deliver.
4. Collaborative work: The three of us worked on the project without meeting physically and still communicating requirements through diagrams. This helped us in gaining hands on experience on how the actual collaborative team functions in the industry
5. Technical Tool: Lucid Charts is a great modelling tool which was not known to us prior to working on this subject. It promotes team-work in a real life environment. With the power to add as many objects as required, the tool can be used in a complex scenario and hence highly scalable.